导读｜入门Python的必备知识

2021-02-03 尹会生

Python自动化办公实战课 进入课程 >

讲述：尹会生
时长 22:31 大小 20.63M

你好，我是尹会生。

咱们这个课程是用 Python 解决办公低效问题，但是**即使你从来没有用过 Python，甚至没有任何的编程语言基础，也完全可以学会这门课。**

为了解决你的语言问题，我特意准备了这节课。我会给你讲解 Python 的五个最基础的方法知识，包括运行环境配置、变量、数据类型、控制语句和使用函数库的方法。

这节课的内容也不需要你马上掌握，其中提到的一些关键知识，我会在后面的课程中详细讲解，包括它们的使用场景和具体用处。**学完这节课，你只要能对 Python 有一个初步的了解，可以看懂基本的 Python 代码就行了。**

当然，如果你有一定的 Python 语言基础，那么这节课就相当于给你巩固复习了，帮你查漏补缺。

运行环境配置

我们先从 Python 的运行环境配置开始说起。初学者面对的最大难题，就是如何让自己的 Python 程序运行起来。

一般情况下，运行的 Python 代码会被保存到一个以.py 作为扩展名的文件中，也就是 Python 脚本文件。要想让 Python 程序运行，我们需要打开终端应用程序（在 Windows 中开始 - 运行 -cmd.exe）：

复制代码

```
python3 /py文件所在的路径/xxx.py
或
cd /py文件所在的路径
python3 xxx.py
```

这段代码中，xxx.py 就是我们编写好的 Python 脚本文件，Python3 是 Python 脚本文件的解释器，它会把我们编写好的代码翻译给计算机，让计算机去执行。

在运行程序的过程中，有两点需要注意。

第一，Windows 和 macOS 默认是没有安装 Python 解释器的，所以你需要通过 Python 官方网站下载不同操作系统下的 Python 解释器。关于 Python 的版本，我建议你使用官方建议的最新稳定版本，下载后直接安装就可以使用了。

第二，Python3 解释器需要在字符终端下运行。

以 macOS 系统为例，你需要打开终端来运行`python3 -V`命令来验证 Python 解释器是否安装成功，如果显示了 Python 解释器的版本，那说明安装是成功的。如果提示没有找

到这个命令，你需要重新安装，并注意安装过程中是否有错误提示信息。

```
python3 -V   # 执行的命令
Python 3.9.1 # 返回的结果
```

搭建 Python 的开发环境，是运行一切程序的基础。无论你现在手边电脑操作系统是哪一种，我都建议你马上行动，先按照刚刚讲解的步骤，正确地安装和设置 Python 开发环境。这样在之后的课程中，你就能马上跟着我的讲解进行操作了。

变量

在正确配置好 Python 的开发环境之后，咱们继续学习一个重要概念：变量。

变量这个词来源于数学，在编程语言中，变量主要是用来存储计算结果或表示值的抽象概念。

计算机中的变量都有一个名字，我们把它称为变量名。这个名字一般是根据变量的英文名进行人为命名的，一般多用简短且易于记忆的名字。比如，我在课程中会使用 src_path、dst_path 两个变量表示操作源文件路径和目标文件路径。

我们还可以为变量指定一段数据，这段数据称作变量的值。计算机处理的数据很大一部分来源于用户输入的数据和运算的结果数据，如果这部分数据需要跟踪，我们就可以将数据存储在变量中。

对于初学者来说，变量的概念比较抽象，你可能现在有点蒙。别急，我现在给你举一个简单的例子，你一看就明白了。假如我要在 Python 当中计算两个数的和，代码如下：

```
x=100
y=200
z=x+y
print(z)
```

在上面的代码当中， 我们分别定义了 x、y、z 三个变量。 x、y、z 称作变量名，定义变量名必须要遵从以下四个规则：

1. 变量名只能是字母、数字或下划线；
2. 变量名的第 1 个字母不能是数字；
3. 变量名也不能是 Python 的关键字；
4. 变量名是区分大小写的。

上述代码中的 = 和 +，称作运算符。

“=”叫做赋值运算符。它的功能是将右边的表达式赋值给左边的变量。

“+”叫做算术运算符。进行算术运算时还可以使用 -、*、/、% 等常用算术运算符号。

代码的最后一行，为了能让你的计算结果在终端上显示，我们需要使用一个叫做 print 的内置函数。

内置函数是 Python 内置的一种功能。比如代码中的 print 就是用来进行输出的。它的最基本用法就是在 print 函数的括号中写入变量，除了输出最终的结果外，也可以将 print 函数放在程序运行的过程中，方便你输出更丰富的调试信息。

通过上面的讲解，我为你介绍了变量如何进行赋值和如何输出。其实，在 Python 中变量用来存储的数据还会区分类，这些类型也有专门的名称，叫做数据类型。

数据类型

Python 支持的数据类型有数字、字符串、元组、列表、集合、字典 6 种，不同的数据类型用于描述不同的类别，比如姓名可以用字符串类型，性别可以用布尔型，身高可以用数字类型。下面的代码，我为你演示一下三种最简单的数据类型：

复制代码

```
1 name = "yin huisheng"
2 male = True
3 hight = 180
```

上面的三个变量，我分别为它赋予了字符串、布尔类型和浮点型这三种数据类型。 Python 的变量不需要事先定义变量类型，跟其他语言相比更加简洁。Python 可以直接对变量进行赋值，这样就可以根据变量值自动识别变量的类型了。

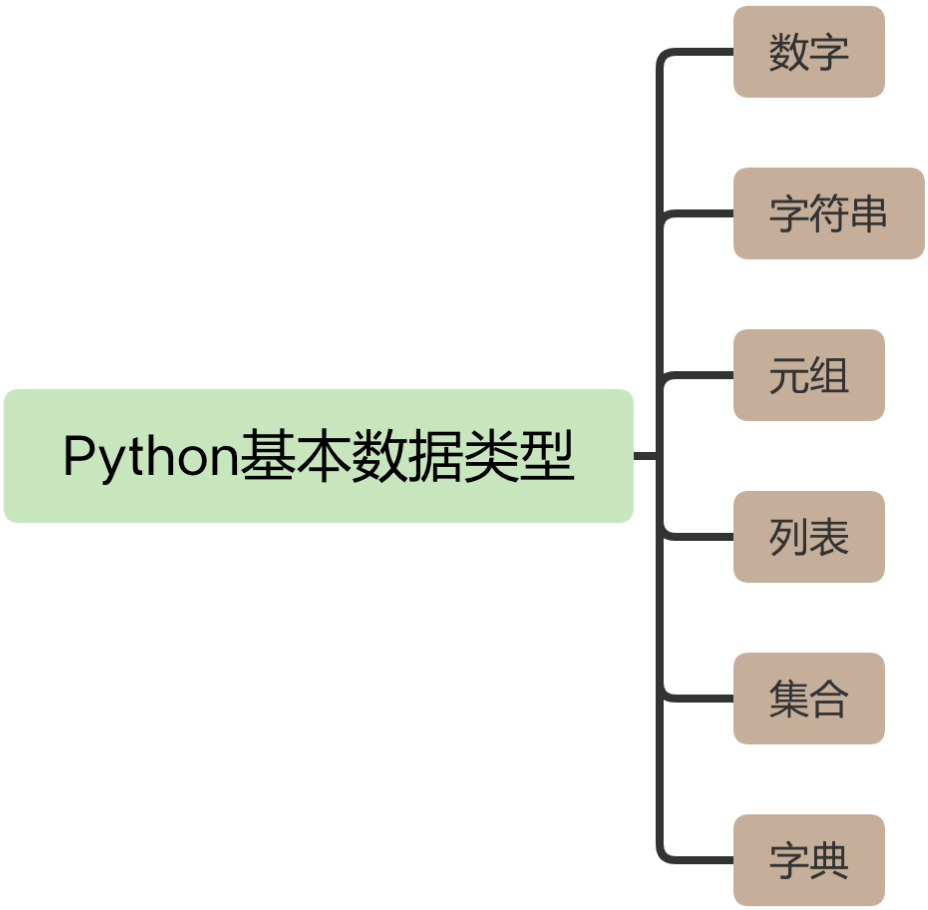
不过在上面的例子当中，你还需要注意两点。

一方面，如果你使用字符串类型，就必须给字符串增加引号。这是字符串的语法规定，如果不加引号会被识别为其他类型，在进行操作时就会得到不同的结果。

比如说，我定义 Python 的变量为“123”，加了引号会被识别为字符串，如果一个变量“123”没有加引号，则会被自动识别为数字。如果在 Python 程序中使用“123+123”会得到两个整数的和“246”，如果是“123”+“123”就会得到两个字符串的连接“123123”。

另一方面，使用布尔类型的时候，True 和 False 首字母必须要大写，它们是 Python 的关键字，如果首字母没有大写会被当作变量处理，解释器也会报错。True 在布尔类型表示真（对），False 表示假（错）。比如算式 10>5 是正确的，在 Python 中就会用 True 来表示结果是真，反之为假。

上面提到的整数、浮点数和布尔型都是属于数字类型。除了这些，Python 还支持很多种数据类型，包括数字、字符串、元组、列表、集合、字典等。它们之间的关系，我用下面的思维导图为你进行展示。



我在课程中会逐一为你剖析它们的用处，让你能够全部掌握。当你学会这些类型的时候，可以更精确地描述数据，比如，我想用 Python 的数据类型存储“5 个苹果和 10 个橘子”，我们就需要使用字典和字符串来表示这些数据。你可以看一下下面这个记录水果数量的例子：

复制代码

```
fruits1 = {"apple":5, "orange":10}
fruits2 = "apple,5,orange,10"
```

我分别用字典和字符串记录了水果的名称和数量，可以看到，字典表达的水果和数量之间的映射关系更明确，而且不同的数据类型还能支持它特有的内置方法。例如上面定义了水果的字典数据类型，我们可以通过 fruits["apple"] 直接得到苹果的数量 5，有时候，这些便捷的内置方法也是我们选择数据类型的依据。

数据类型是一门编程语言的必要组成部分，学习编程语言主要就是为了掌握它的数据类型。这节课我只要求你能够了解 Python 有哪些基本数据类型就可以了，在之后的课程当

中，我会为你重点介绍每一种数据类型的优缺点，以及该怎么使用它们。

流程控制

流程控制，就是我们经常说的程序执行的顺序控制。 从上面的几个例子当中，你会发现程序都是从上向下依次执行的。这也是 Python 语言最简单的结构——顺序结构。在 Python 中，除了顺序结构之外，还有分支和循环两种结构。

分支结构

分支结构用来判定某个条件是否成立，它会根据一条或多条语句的判定结果（是 True 还是 False）来执行对应操作的语句，从而实现分支的效果。实现分支结构的语句叫做 if 语句。除了 if 这个关键词外，我们经常还会用 if...else 和 if...else...if 语句来实现更多的分支和更复杂的逻辑。

我们一起看一个比较数值大小的例子吧。

复制代码

```
x = 20
if x > 10:  # 注意结尾的冒号:
    print("if的判断结果为True")
    print("x的值大于10")
else:
    print("if的判断结果为False")
    print("x的值小于10")
print("程序执行结束")

# 执行结果如下
if的判断结果为True
x的值大于10
程序执行结束
```

在上面这段代码当中，我们实现了判断变量 x 是否大于 10 的功能。在书写分支结构时，要注意书写格式问题，分支结构的书写格式比顺序结构要复杂，例如：

1. if 判断的书写格式，冒号标志着 if 判断的结束。
2. if 下方有缩进的代码，是在 True 的条件下才能运行的。在演示代码中，缩进代码由 2 行组成，其实，它也可以由很多行代码组成，多行代码的形式被称作代码块。代码块一

直到缩进再次改变的时候，True 条件下的代码才结束。

需要你注意的是，在整个程序当中，缩进方式必须一致，要么使用 Tab、要么使用 4 个空格进行缩进。Python 对于缩进和复合语句结尾的冒号要求非常严格，如果弄错了就会报错。而且，缩进不同会大大影响一个程序的运行结果。例如上面的代码中“print("程序执行结束")” 如果有缩进，你就看不到“程序执行结束”这行执行结果了。

循环结构

在程序运行逻辑中，另一个重要的结构叫做循环结构。**循环结构是可以多次执行同一段代码的语句结构**。在 Python 当中有两种循环语句，分别是 while 语句和 for 语句。

while 语句在循环开始前先对条件语句进行判断。如果条件语句的结果为 True，则循环一次再次进行判断。如果条件允许的结果为 False，则循环结束。这里所指的条件语句和 if 语句当中的条件是一样的。下面的例子是我使用 while 语句实现的从 1~10 的累加。

复制代码

```
start = 1
total = 10
sum = 0
while start <= total: # 条件为True 继续执行循环
    sum += start
    start += 1
print(sum)  #条件为False循环结束,输出结果
```

另一个循环语句叫做 for 语句，它和 while 的功能有所不同。我们一般把 for 的功能称作遍历功能。一般我们会提供一个列表，通过 for 将列表中的每一个元素读取出来，进行处理。

举个例子，如果我们要依次输出列表中的每一个元素，就可以用下面的代码：

复制代码

```
total = list(range(1,11))  # 定义一个1-10 的列表
for i in total:
    print(i)

# 输出结果
1
2
```

```
3
4
5
6
7
8
9
10
```

流程控制语句可以改变 Python 程序的执行顺序，而我们通常会根据实际的业务场景来控制某些语句块进行重复执行的。

使用函数库

Python 语言的强大之处，除了语法简单外，就在于它庞大的函数库了。

Python 的函数库有标准库和第三方库（也叫扩展库）。标准库是 Python 自带的，而扩展库是需要再进行安装的库。你可以从[Python 的官方文档]((https://docs.python.org/zh-cn/3.9/library/index.html) 和PyPI 网站中找到它们，其中，PyPI 网站的库就是扩展库。

PyPI网站上面的项目超过 28 万个，几乎包含了各个行业和各个功能的应用，实现了很多你日常需要手动操作的功能。

这些库如何是用呢？首先看一下官方自带的库。

如果要使用官方自带的库，可以使用 import 关键字导入。例如我计算算数平方根功能，就可以使用如下方法：

复制代码

```
import math
math.sqrt(16)
```

import 后面指定要使用的库的名称，使用库的具体功能，通过官方文档来找到相应的函数。例如，算数平方根使用 sqrt 函数就可以实现你想要的功能了。

标准库的使用非常容易吧，那如何使用扩展库呢？由于扩展库都需要通过网络进行下载，因此我们还需要提前**在终端使用 pip3 命令进行下载和安装**。例如我希望 Python 可以支持 Word 文件格式, 我就需要在终端执行下面的代码。

```
1 pip3 install python-docx
```

复制代码

下载和安装之后, 就可以像标准函数库一样使用 import 导入使用了。

总结

通过上面的内容，我为你简单介绍了一下入门 Python 需要掌握的五大部分，包括环境配置、变量、数据类型、控制语句和使用函数库的方法。这五个部分是 Python 最基础也是最必要的语法知识。希望你能通过我的介绍，对 Python 语言有所了解，消除对编程语言的畏惧心理。

最重要的是，即使今天这节课你没有完全懂，也没有关系。

一方面，我会在之后的课程中不断讲解每个部分的用法和具体使用场景。还是我在开篇词中说的学习方法，你需要在听我讲解的同时，动手操作，反复练习，这些语法知识就不会是太大的问题。

另一方面，我也希望你能在学习之后的课程时，如果遇到哪些关键语法知识不太明白，就回头来学习今天这节课。这节课相当于整个课程的基石，需要反复琢磨、反复学习。

如果你能跟着我的课程不断练习实操，那学完后，你都能够熟练地修改课程中涉及到的代码，并且还能根据自己的工作场景，自行编写提升工作效率的程序。

思考题

最后留一道思考题给你，你以前是如何学习编程语言的呢? 有没有好的学习方法分享一下? 或者你在学习过程中有遇到什么困难，也可以分享出来。

如果你觉得这节课能帮助你提升办公效率，欢迎点击“请朋友读”，分享给你的朋友或同事。

提建议

© 版权归极客邦科技所有，未经许可不得传播售卖。 页面已增加防盗追踪，如有侵权极客邦将依法追究其法律责任。

上一篇 开篇词 | 重复工作这么多，怎样才能提高工作效率?

下一篇 01 | 拆分与合并：如何快速地批量处理内容相似的Excel?

精选留言 (13)

写留言

Fan
2021-02-03

重要的事情说三遍，多看官方文档，多看官方文档，多看官方文档。会让你少走很多弯路。

作者回复: 官方文档很重要。但是对于刚接触Python的人直接看官方还是有点困难的。可以在课程过半的时候再来看。理解会深入一些。

5

公号-技术夜未眠
2021-02-08

一门新语言的快速入门指北:
鸟瞰全貌
对比之前已经掌握过语言间的差异
写一个简单的demo，先运行，后一步步调试
写一个小的项目，比如爬虫程序...
展开

作者回复: 加油

1

落曦
2021-02-07

基础入门，这一篇就够了，开心，复习了！
学编程 知行合一 多写代码就可以啦

作者回复: 加油

1

linker
2021-02-04

学c语言，先照着书上敲代码。

展开

作者回复: 那样效果不好。配合专栏问题加课后代码服用，再。照着书上。系统性的过一遍，理解的更深刻。

1 1

朦胧银月
2021-02-17

基础入门，我喜欢搜索"xxx the right way"的入门指南文档，跟着入门指南熟悉基础知识和进阶路径，后面实战敲代码做demo，遇到问题查阅官方文档，当然还有stackoverflow

Cherilyn
2021-02-15

请问有视频和课件吗？谢谢

展开

Cherilyn
2021-02-15

请问有视频和课件吗？还是只是音频？谢谢

展开

编辑回复: 有文字稿和音频，还有老师给的可供运行的小程序。这些已经可以把内容讲清楚了，所以没有视频哦～

海的对岸

2021-02-09

边看便敲，碰到问题，排查问题，上网查
编程的学习过程，在我看来，就是一条缓慢的学习曲线，很长一段时间，成效看起来比较低，但是你学着学着，之前学的，慢慢就都理解了，然后就会了

作者回复: 还差了一个步骤,总结并优化
代码要归类整理好,随着不断开阔眼界,要持续迭代自己的代码

Soul of the Drago...

2021-02-07

我现在在学习爬虫，感觉学了很久，自己还是不能写出一段完整的爬虫代码。

作者回复: 试试和自己的生活、工作结合起来,不追求全自动话,先解决单个数据获取问题

McKee Chen

2021-02-07

留言对我来说，不仅是打卡，也是本课程学完后的总结，还能跟评论区的同学们之间进行互动和交流，很有趣~

展开

作者回复: 加油了

McKee Chen

2021-02-07

一开始我是看书学习最基础的python语法，等基础语法掌握了，再跟着课程进行深入学习和实战，学习新知识的同时也能复习以往学过的知识。
最大的感悟还是多上手，多思考，多交流，把自己学到的跟朋友或一起学习的人进行交流与探讨，不仅加深自己的印象与理解，还能提高编程熟练度。
相信跟着尹老师学习会有很大的进步，给自己加油，也给同时在学习本课程的同学加油~

展开

作者回复: 多写多思考,勤看文档,出现问题多问几个为什么,一起进步

建明
2021-02-06

老师，我们公司用的是内网，和外界沟通需要用专门的保密U盘，这种不连接互联网可以使用吗。

作者回复: python可以下载到u盘中进行安装，但是很多网络的自动化功能就没办法使用了

1

2021-02-04

刚好学到了老师零基础入门python课程的while 和 for循环语句，看这一节就比较容易理解。
两门课可以同时进行。

展开

作者回复: 一个实操，一个办公场景加深理解